Towards Open Telco – Business Models of API Management Providers

Yrjo Raivio  
Aalto University, FINLAND  
yrjo.raivio@tkk.fi

Sakari Luukkainen  
Aalto University, FINLAND  
sakari.luukkainen@tkk.fi

Saku Seppälä  
Aalto University, FINLAND  
saku.seppala@tkk.fi

Abstract

The term Open Telco refers to a new phenomenon where telecommunication operators provide open APIs for 3rd party developers and content providers. Multi-operator support requires cross network service providers (CNSPs) or network brokers between operators and developers. In the mobile domain brokers have already stepped into the SMS and data roaming businesses, but API brokering is still an emerging phenomenon. Business models based on mobile brokering are also in an early phase. On the other hand, internet API management providers have been offering services to various API publishers for a long time. This paper utilizes internet case studies in order to find out the best mobile broker business models. The analysis is carried through using the STOF model. At the end, the results are verified with the literature review, and recommendations for mobile brokering and for the next steps are given.

1. Introduction

Since the launch of GSM in 1991, Finland has been one of the forerunners in mobile voice and text messaging services. However, the saturation point of the core mobile services was reached fast, and the average revenue per subscriber (ARPU) started to decline. One reason for this development was the introduction of a number portability feature that was realized in 2003. Increased competition has resulted in declining user loyalty and increased customer churn. Finnish authorities also intervened to guarantee equal network usage fees to all service operators.

At the beginning of March 2004 network operators cut their fees to service operators by approximately 30 percent. Diverse new entrants, called Mobile Virtual Network Operators (MVNOs), emerged in the market. Competition between the operators became price-based, ARPU decreased significantly from 40 euro to less than 30 euro. All operators had a similar service offering, and in addition to voice and SMS, high roaming fees were significant revenue sources. Simultaneously, the growth of mobile data services was still very modest [1]. Disappointment in WAP and MMS slowed down the data usage, but on the other hand, the introduction of flat rate data subscriptions and new smart phones turned the data curve in a positive direction. Parallel to that the incumbent operators started to make acquisitions - MVNOs disappeared, price competition settled down, and incumbent operators started to increase prices.

The major mobile telecommunications operators have been used to operating according to a walled garden business model. In this model operators fully control applications that are available to their subscribers. As a result, innovativeness remains at a low level, service development is slow and prices are high. Bundling of equipment, subscription and services helps 3G adoption rates, but it also promotes a walled garden business model, as tailored service packages increase switching costs. At the same time, the Internet and telecommunication deregulation has started to disrupt the walled garden approach, leading towards more open models.

Continuous experimentation is one of the secrets of the Internet success stories. Open APIs enable open innovation by attracting developers to work on the core assets. New services are created, tested and verified within short cycles. Service innovation is essential for mobile industry, too. New mobile services are needed to increase market segmentation based on open innovation and the Open Telco [2] approach. The key issue of promoting service innovation is achieved by lowering the usage and development barriers by arranging a suitable experimentation infrastructure with reasonable cost structure and openness. Also mobile operators have realized the challenge. GSM Association (GSMA) is currently running a new initiative, called OneAPI to enable mobile APIs [3].

1.1. Research approach of the study

The literature study will analyze the business drivers for an increased interaction and collaboration between companies, which will help to understand the underlying factors for the increased interest in providing open APIs. Topics that are covered include business ecosystem development, experimentation as a
means for managing market uncertainty, open innovation and two sided platforms. The focus is in the developed markets.

Open APIs are still rare in the mobile domain. SMS and data roaming brokers have existed a longer time, and in 2010 GSMA started a national pilot with the OneAPI standard in Canada [4]. The results of this pilot are expected towards the end of 2010. Due to a lack of mobile API data, this paper also studies the business models of internet API management providers, namely Mashery, StrikeIron and 3Scale, in addition to the preliminary results from the OneAPI pilot. It is worth noticing that mobile network APIs provide similar data to internet APIs, meaning that the results from the internet cases are applicable to mobile cases as well. In addition, mobile applications usually utilize both OneAPI and internet APIs.

The research was carried out using a multiple-case study research methodology. Multiple-case designs enhance the external validity, because research methods such as replication logic can be used to test the generalization of the results. Yin [5] observes that each case should be treated as a single experiment, and the analysis must follow a cross-experiment design where each case confirms or disconfirms inferences drawn from the others. If similar results are obtained in several cases analyzed sequentially, a replication will have taken place. The business models of case studies are analyzed using the STOF (Service, Technology, Organization, and Finance) business model domains and, partly, the STOF method [6]. The full STOF method includes four steps, but in this paper only the first step, Quick Scan, is applied.

The structure of the paper is the following. Chapter two includes the theoretical background, followed by industry drivers for openness in the mobile domain. Chapter four presents the case studies, chapter five discusses the mobile broker alternatives, and finally chapter six summarizes the results.

2. Theoretical background

2.1. Business ecosystems

Business ecosystems offer an alternative approach to value networks, which also began to appear in academic papers in the beginning of the 1990s. It is good to acknowledge that in the literature there are basically two slightly different approaches to business ecosystems.

The first approach is to consider the whole economy as an ecosystem. As Michael Rothschild [7] states: “A capitalist economy can best be comprehended as a living ecosystem. Key phenomena observed in nature – competition, specialization, co-evolution, exploitation, learning, growth, and several others – are also central to business life”.

The second approach is to keep business ecosystems as a smaller group of companies that work together on a certain technical platform or produce a certain service. Moore [8] defines business ecosystem as a group of companies that “work co-operatively and competitively to support new products, satisfy customer needs, and eventually incorporate the next round of innovations”. Moore’s ecosystem concept was widely used by Cisco Systems Inc, which created a business ecosystem of business partners around its core products to expand sales, manufacturing and technical support. This strategy was highly successful for Cisco and made it the most valuable company in the world for a short time in 2000, but it was then hit hard by the 2002 technology bubble crash.

Peltoniemi [9] has done an extensive synthesis on different interpretations of ecosystems and their properties in the context of business and economics literature and has created a diagram of the characteristics of a business ecosystem, presented in figure 1.

![Figure 1. Characteristics of business ecosystem [9]](image_url)

Based on Peltoniemi’s synthesis, the business ecosystem is a dynamic structure, which consists of an interconnected population of organizations that has unclear borders. An important aspect is that business ecosystems are also coupled with a changing environment which can lead to unpredictable dynamics in the ecosystem. In value networks the relationships between companies are more stable and predefined, whereas in ecosystems the interactions of companies are more dynamic and companies have a low barrier to joining or leaving the ecosystem.

Ecosystems simultaneously contain cooperation and competition between its members, but at the
population level the interconnectedness leads to a shared fate in the sense that the failure or success of one company may echo through the whole system, which forces companies to seek a balance between their own benefits and the health of the entire ecosystem. Business ecosystem thinking assumes that organizations have conscious choice that aims at delivering innovations and commercial success.

Brandenburger and Nalebuff [10] made an important finding that companies should put more effort into innovative strategies. As a result, win-win strategies will be possible, because the win-lose strategies of capturing new market with lower prices is often counterproductive. They suggest long term cooperation, to find win-win as well as win-lose opportunities. They credit the use of the term to a former CEO of Novel, Ray Noorda, who has used it to describe relationships in the information technology business: "You have to cooperate and compete at the same time”.

2.2. Market uncertainty and experimentation

Gaynor [11] defines market uncertainty as “the inability of vendors and service providers to predict what the users will like”. Market uncertainty varies between different industries, depending on how mature and stable they are. Market uncertainty is highest among emerging technologies, but it becomes less as their markets mature. After a dominant design is established, market uncertainty can once again reassert itself when new disruptive technologies are introduced. To manage market uncertainty, Gaynor suggests that firms should carry out experimentations to find out which product features are most valued by the customers. Gaynor also argues that experimentations are most effective when the market uncertainty is high and less valuable when market uncertainty is low.

According to the normal product cycle, companies release new, improved product versions based on the feedback gathered from the previous products. However, it is difficult to organize experimentations inside a company. An effective setup would require several parallel development teams. Independent teams would mean duplicated efforts, which would not be financially very efficient. This is a reason why there is a need for activating external developers to create complementary applications for the experimentation process.

2.3. Open Innovation

Open innovation is a term that has been made well known by Chesbrough [12]. Chesbrough concludes: “Open innovation is a paradigm that assumes that firms can and should use external ideas as well as internal ideas, and internal and external paths to market, as the firms look to advance their technology”.

Open innovation is in contradiction to the old way of managing innovation and business in large corporations that Chesbrough calls closed innovation. According to the closed innovation paradigm, companies must generate their ideas by themselves, and then independently develop, market, distribute, finance and support the ideas further. This leads to highly vertically integrated corporations such as, Xerox, IBM and GE. Vertical integration describes either development into activities involved with the inputs of the company, backward integration, or development into activities involved with the outputs of the company, forward integration [12].

In opposition to the closed innovation paradigm, the open innovation encourages corporations to let ideas and research projects traverse company boundaries as illustrated in figure 2. According to the paradigm companies should actively search for ideas for new research projects from external sources, such as universities and other firms. This will allow the company to tap into a much larger pool of new ideas, than just the ones that are created internally, and maximize the company's potential to create new commercial products.

![Figure 2. Open innovation](image-url)
Open API is a good example of the open innovation principle. Instead of trying to restrict customers from accessing services, many firms have released open APIs on their web-sites. Open APIs attract third party developers into becoming an asset to the company, and as a result, either new versions or totally new services will be deployed. One of the best examples in this category is Flickr, which was invented as a side project of the original online game service. The variety of services available to the customers is not anymore limited to the innovation potential inside the company. The more companies release open APIs, however, the more there is need for API management and brokering services.

2.4. Service ecosystem

Barros and Dumas [13] define a service ecosystem in terms of five actors: provider, user, broker, mediator and specialist intermediaries. Services are offered by the providers and consumed by the users. A broker connects service providers and users together offering additional services, such as a single sign-on and payment, in one service offering. A mediator provides service adaptations for different data formats and service functions, allowing brokers to concentrate on their core business. A specialist intermediary offers specialized service components for providers. These can be, for example, monitoring and payment services.

In addition to the above, Riedl et al [14] mention that there are platform providers who offer an overall platform for other providers to work with. Amazon Elastic Compute Cloud (EC2) and Force.com are the most successful examples of those services. Furthermore, Riedl et al [14] propose a collaboration model with four actors connected to an innovation space as shown in figure 3.

![Figure 3. Service ecosystem [14]](image)

Ballon [15] defines a two-sided network theory that describes cases, where two types of customers interact on a platform. The interaction is affected by indirect network externalities locating on opposite sides of the platform. According to Evans & Schmalensee [16], many diverse industries are occupied by businesses that operate two-sided platforms. The customers of these businesses have a relationship between each other, and for that reason, a common meeting place is required. Two-sided platforms are common in old, advertising based economies, and they play an important role in minimizing the transaction costs between the customers.

Furthermore, Evans & Schmalensee [16] define five critical factors for the size of two-sided platforms. The list includes indirect network effects, scale economies, congestion, platform differentiation and multi-homing. Evans and Schmalensee also note [17] that brokers can act as catalysts best when both sides of the market are equally balanced. If there are too few or too powerful players on either side, a broker cannot function effectively. Finally, the authors warn that a few cautions must be added to the theory on two-sided platforms. First of all, the industries have been defined with abstract models that might not work in the real life. Secondly, there is a lack of empirical data on two-sided platforms, and thirdly, the theories presented depend very much on the industry area.

Evans & Schmalensee [17] list four types of two-sided platforms: matchmakers, audience-makers, transaction based and shared input. Eisenmann [18] divides platforms into two categories: proprietary and shared. The proprietary system, such as Google, has one single provider that fully controls the technology. In the shared system, multiple companies develop the common platform. The Linux operating system is an example of this approach. According to Bambury [19], most internet services are disinter-mediated, because that is the cheapest solution. However, most real-world businesses are intermediated. Bringing real-world products onto the Internet will inevitably increase the need for new intermediate layers. This development may be positive for consumers, but it creates challenges for real-world profit margins.

3. Industry drivers

3.1. Open Telco

The Open Telco [2] term refers to a state where operators seek an optimum operation model between closed and open models. In the past, most of the incumbent mobile operators that were often owned by a government, operated according to the closed, so-called, walled garden model. Typically, an operator controlled all actions in the closed model that was built
on top of standardized interfaces, protocols and services. The business model was simple: subscribers paid to operators a transaction fee per services used. Most of the services were created by the operators themselves or by their partners, to whom operators paid a minority commission.

At the other extreme operators can select an open model, also called a bit pipe. This model is common among Internet Service Providers (ISPs), but also a few challenger and MVNO operators have chosen the open alternative. The main idea is to provide an efficient transport solution for the customers, but also to leave the service layer open for any player. End users have full access to any service provider content. Operators often charge for the bits on a flat rate basis. The variation in the services is large, but on the other hand, the service quality can vary and security challenges may exist. For an operator the open model does not provide a good business incentive due to the flat rate charging system. Besides, the lack of control will lead to fierce price competition, where operators do not have the tools to differentiate from their competitors.

Instead of the extremes, operators can also select a hybrid model. NTT DoCoMo’s iMode-service was one of the first examples where the operator applied experiences both from the closed and open models. The hybrid model combines the best parts of both extremes. The main idea is to utilize the operator assets but embrace the benefits of openness. Open Telco is just one example of the initiatives that support the hybrid approach. Text messaging can be already marked with a hybrid label, because it allows cross operator innovations, but on the other hand, the business models are still operator specific. Figure 4 shows examples of service, technology, organization and finance areas mapped into different mobile models [20].

Many mobile operators have started independent developer trials with specific APIs. The most common APIs are related to messaging, location and payment but also other APIs have been seen. A large variation creates a problem for developers, because services will not work without modifications across the operator boundaries. GSMA realized the challenge and started a standardization activity to harmonize the most common APIs [3]. As a result, the first official version of the OneAPI specification, including messaging, payment and location APIs, was published on June 2010. GSMA has also planned to continue the specification work, and the second version of the OneAPI specification should follow later in 2010.

3.2. Towards brokering

Social media services, such as Facebook, Twitter and YouTube, are becoming increasingly popular and changing the way that people use the Internet. At the same time, iPhone has accelerated the usage of the mobile internet and created a real demand for 3G mobile networks. Due to new players, however, market uncertainty in the ICT space is very high.

With open APIs, companies can harness the power of third-party developers for creating new, experimental services on top of their platforms. New features will be added to the original services, and the popularity of new versions will be immediately tested. Based on the feedback, the most successful features can be integrated into the main service. The process benefits both the original service owner and the open community. In future, this development will drive more companies into open up APIs to their services, which will in turn create more demand for API management and brokers.

The concepts of web 2.0 and mashups have had a strong influence on the success of the open internet APIs. Web applications promote user interaction and combine data from various sources to create novel and innovative services. Several web sites and businesses already now provide APIs to 3rd party developers. As a proof of this, for example, Twitter API receives ten times more traffic than the actual website. A huge amount of API traffic requires efficient API management procedures to prevent unfavorable behavior of developer applications. It is clear that API providers must control the traffic. For example, the social networking site Twitter has restricted the number of API calls per application to 20 000 per hour.

API management is also required for connecting cross network service providers (CNSP). Brokering evolution from an Internet based solution into a network broker model can be expected. The bit pipe model utilizes the Internet as a transport solution bypassing the operator assets. In the second phase, each operator makes bilateral agreements with ISPs,
which creates a complex mesh network. Large operators and ISPs may be able to handle the complexity, but for most companies the system is unmanageable. The ultimate goal is shown in the last phase. The network broker combines the best parts from the previous scenarios. Both operators and ISPs have a simple interface with the broker, and simultaneously, the internet players can also utilize the operator network assets. Figure 5 presents the ultimate step. [21]

![Network broker model][21]

4. Case studies

4.1. Mashery

Mashery was founded in 2006 and their web API management service was launched in the same year. Of all of the internet case companies, Mashery has the most extensive and impressive client list that includes many widely known companies such as Netflix, LinkedIn, The Guardian, New York Times, Best Buy and Shopping.com.

Mashery provides a full API management service that requires practically no changes in the existing service delivery framework, which makes it an easy solution for opening already implemented private APIs. The API management features consist of sign-up, enrollment, subscription and quota management, detailed usage metering and developer support. From the web service consumer’s point of view, the service is basically a fully customizable developer information website for assisting developers to building applications using the API. For the API provider, the website has management interface for managing the API services and the users, viewing usage logs and setting restrictions.

It is no coincidence that Mashery’s API management service was launched the same year as Amazon EC2 cloud computing platform. Mashery has been Amazon’s customer from the first day. By utilizing cloud computing services, Mashery has been able to get rid of the preliminary investment burden, and instead, they have been able concentrate on developing their core services. Technically Mashery’s API management provider solution operates as a transparent proxy between the API users and providers. Organizationally speaking, the value network is simple because API users and providers are connected to Mashery, which in turn runs on Amazon’s cloud infrastructure.

Mashery charges its customers a predetermined monthly fee, which is based on estimated usage volumes. At the time of writing, Mashery does not list on their website any example rates, but prices are negotiated individually for each customer. Mashery pays Amazon for its cloud services only based on the actual computer-hours used, which in fact eases Mashery’s own cost and profit calculations. In addition, Mashery can easily scale up their operations when the service becomes more popular, because it can dynamically acquire more server resources from Amazon if needed without making heavy investment decisions.

4.2. StrikeIron

StrikeIron’s web service marketplace was first announced in 2005. StrikeIron’s service differs from other case study companies mainly on two respects. StrikeIron has true market place orientation and heavy reinforcement of their brand. The clear orientation towards a true market place experience can be seen in the product design. Web services are listed on the web site divided into different categories, such as communications, financial, marketing and others. For developers the categorization enables easy integration of new features into their own applications.

About half of the web services available through the market place are produced by StrikeIron while the other half are produced by various third party companies. The success of StrikeIron has been modest if the number of different web services is the measurement tool. The number of web services actually dropped from 50 to 35 between the years 2005 and 2009. Instead, StrikeIron has concentrated on offering data services for specialized areas, such as an online Customer Relations Management (CRM) solution provided by SalesForce, rather than becoming a general all-purpose web service market place.

The technological architecture of StrikeIron’s service implementation is basically identical to Mashery’s solution. They both use the proxy principle, and pass API calls between the users and the API providers. The only difference can be found in the platform implementation. While Mashery utilizes
Amazon’s EC2, StrikeIron has chosen a private cloud alternative, called IronCloud. It is notable that Amazon’s cloud computing service was not available during the time when StrikeIron was launched.

StrikeIron’s value network is a little more complex than Mashery’s. StrikeIron has to manage relations with hardware providers and internet operators in delivering its service. The private cloud infrastructure together with the proxy type of service will make it more difficult to expand the service, especially on a global scale. To keep the API access latencies low and to ensure a high availability of service, StrikeIron has distributed its service infrastructure to several internet operator networks.

StrikeIron’s primary service model is directed towards brokering enquiry based web services, such as address verification and credit checks. As a result, the revenue model is based on charging a commission on the API provider’s income. The subscription models are quite similar to Mashery’s models, and are based on annual or monthly prepaid subscription fees, with a few different predetermined quota options for each. Because StrikeIron utilizes its own private cloud infrastructure, it has to carefully plan the future needs for expanding its server and network infrastructure. This may be one reason why StrikeIron has concentrated on offering commercial APIs where the usage volumes are more stable and easier to predict.

### 3Scale

The third internet case company, 3Scale Networks was founded in 2007. After two years of operation, they also launched an API management service. Compared to StrikeIron or Mashery, 3Scale Networks does not offer any additional service features. The product features are basically the same as with the competitors: sign up management, subscription management, usage tracking, quota management, support website for online documentation, forums, blogs, and billing.

The biggest difference between 3Scale’s and the other API management services is in how 3Scale’s service combines web service market place functionality with individually customizable API support pages. 3Scale offers a service package with various API management brands, combined with the market place and billing support. In addition, 3Scale supports both free and commercial APIs.

Technologically the implementation of 3Scale’s API management service differs greatly from the architectures of Mashery and StrikeIron. The difference is most apparent from the perspective of the API providers. While Mashery and StrikeIron provide transparent proxy API services, 3Scale’s service is based on application level software libraries that are plugged into the code of the API implementation. This means that the API traffic is not relayed through the 3Scale platform, and the implementation becomes simpler and more efficient. There is less network traffic and processing, which lowers the service delivery costs. For that reason, 3Scale offers a free usage plan of 50 000 calls per day. As a negative impact, the integration of 3Scale’s libraries into the application code creates a vendor lock-in situation. The value network of 3Scale is identical to the equivalent of Mashery’s.

Due to simpler technical architecture, 3Scale is able to offer its services at a lower cost than its competitors. As in Mashery’s case, 3Scale’s primary revenue model is to sell the service for a monthly fee based on the number of maximum API calls allowed per day. The price range is from 50 000 calls per day up to 10 million calls per day. At the lower limit the usage is free, but a maximum is charged of 1000 dollars per month. In addition, 3Scale offers API providers the opportunity to charge the API users and make a commission out of the income. Similarly to Mashery’s case, 3Scale utilizes Amazon’s cloud services.

#### 4.4. GSMA OneAPI

The GSMA has been leading the open network interface standardization efforts with their OneAPI specification [3]. OneAPI 1.0 version includes definitions for messaging, location and payment APIs, and other APIs will follow later. The first commercial pilot of OneAPI, started at the beginning of 2010 in Canada [4], is based on a broker model, in which APIs for common network capabilities are provided in a multi-operator environment by a cross-network simplification broker, run by the GSMA. The architecture is shown in figure 6.
At first the service offering is limited, but on the other hand, the GSMA OneAPI service roadmap includes extensions depending on the developer feedback. In technology both simple REST and more advanced web service APIs are supported. The GSMA acts as a network capability broker and as a single-contract point for network operators and external developers. The GSMA also takes care of the billing and payment transactions between developers, users and operators. For the developers the APIs have same prices independently of the operator used, but, due to national competition rules, prices between the operators and the broker are secret. SMS sending and location requests have a fixed price and priced per usage. The payment service is based on the revenue sharing model with a 70-30 split, i.e. the developer earns 70 percent of the turnover and the rest goes to the broker and the operators.

The GSMA OneAPI Canadian pilot case differs significantly from the internet case studies. All internet cases are operating in mature businesses where both traffic loads and number of actors are large. The GSMA pilot is in its early phases and no major conclusions can be made yet. However, a few early findings can be listed. In services API selection is limited and more variation should be actively searched for. API specifications have been standardized including a roadmap for future extensions, but performance and security aspects must be evaluated upon the wider deployment of the technology.

From organization’s point of view the broker model is simple. Each operator and developer has only one access point, simplifying the contractual processes. On the other hand, the broker is a new player in the value network requiring its own management. In the early phases of a new service concept the position of the broker can be economically questionable. Finally, the Canadian pilot lacks transparency in respect of the API prices, hindering real price competition between the operators.

4.5. Summary of case studies

API broker 3Scale has the best service mix off the three internet case companies. Their service portfolio includes API management services for both free and fee based API web services. Technologically Mashery’s service can be considered as the optimal approach. It is a proxy type of implementation that is transparent to developers and API providers. As a result it does not require any changes to the existing implementations when new APIs are introduced. Organizationally both Mashery’s and 3Scale’s public cloud based solutions are identical, giving more flexibility than StrikeIron’s private cloud platform. API management solutions can be implemented very effectively on cloud infrastructures, minimizing organizational complexity.

Financially there are three major differences between the services. These factors include the billing options, the ownership of the server infrastructure, and the operational and integration costs resulting from the technical architecture. Mashery and 3Scale use mainly monthly subscription fees that are based on the number of hits the API receives in a month and the required service level agreement (SLA). In addition, 3Scale offers a possibility for fee based web services. On the other hand, StrikeIron charges a commission of the income that it collects from the API users. Because the majority of the APIs are free, and simultaneously the fee based web services are a minority, it is important to offer a monthly based billing option. As the adoption of APIs grows in the future, it may become ever more important to offer the option for fee based web services.

The second important financial issue is whether the service is run on a private or a public cloud infrastructure. By using a private cloud infrastructure StrikeIron has to commit considerable financial resources to server infrastructures and has to constantly plan the future investments to prepare for growth and fluctuations in demand. By utilizing Amazon’s EC2 cloud infrastructure, Mashery and 3Scale can dynamically scale their service based on demand without significant financial commitments. In the long run the private cloud infrastructure can become cheaper, but involves a high investment risk.

Finally, there are operational and integration costs resulting from the technical architecture. Mashery’s and StrikeIron’s proxy solutions require that all API traffic is routed through the API management provider’s server, which makes the service more expensive compared to 3Scale’s implementation. On the other hand, 3Scale’s solution requires that API functions are reprogrammed using the 3Scale’s API management libraries, which implies significant integration costs.

As a suggestion to API management providers, the best way to avoid the high integration expenses, would be to transfer some of the proxy server software functionality to the API provider’s network. The proxy in the API provider’s network would only need to transfer the API management data in a packed format to the API management provider’s servers, which would reduce the required network bandwidth and server resources.

The fourth case study, the GSMA Canadian OneAPI pilot is its early phases, and direct comparison with other case companies is not fair. The service mix is limited and the technology has not been tested under
high load. On the other hand, standardized APIs lower the integration costs. The organization structure has both positive and negative issues. Simple contract procedures favor a broker solution, but on the other hand, a new organization layer adds a new actor to the value network. Financially the broker enables a simple pricing structure, but on the other hand, developers cannot benefit from the price competition of the API prices. See table 1 for a summary of all results utilizing the STOF categorization. The evaluation scale is +, = and -, from the best to worse.

Table 1. Summary of STOF case analysis

<table>
<thead>
<tr>
<th></th>
<th>Mashery</th>
<th>Strike-Iron</th>
<th>3Scale</th>
<th>One-API</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>=</td>
<td>=</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Technology</td>
<td>+</td>
<td>=</td>
<td>=</td>
<td>+</td>
</tr>
<tr>
<td>Organization</td>
<td>+</td>
<td>=</td>
<td>+</td>
<td>=</td>
</tr>
<tr>
<td>Finance</td>
<td>+</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
</tbody>
</table>

5. Discussion

Several theories from academic business literature support the assumption that businesses are becoming more networked. Products and services are built in collaboration, which supports the increasing adoption of open APIs. First, the way the value is considered to be created in companies has changed from a linear model, where each company just adds value to its inputs and passes them to another company, to a complex interaction of value networks and ecosystems, where companies cooperate and compete simultaneously [8, 9, 10].

Second, Chesbrough’s Open Innovation concept has awoken most companies to the possibilities of using external ideas to maximize a firm’s innovative capacity and to profit from innovations that do not fit into the company’s current business model, by finding external paths to market them [12]. In addition, Gaynor’s theory of utilizing experimentation to prepare for high market uncertainty benefits also from open APIs [11]. By releasing open APIs companies can use outside developers to experiment with new product features at low cost. Technical predictions of the future of the internet see the future as an ecosystem of dynamically aggregated and interactive services [22]. The more services need to interact and share data between each other, the more open APIs and API management services are needed.

The first success factor in the service domain is that the service must be easy to integrate into existing legacy systems. This is supported by the success of Mashery’s proxy solution, which does not require any changes in the existing API implementations. In the technology domain, an invisible proxy is a superior choice compared to solutions that require changes in the programming of existing API implementations. As a backend solution public clouds can be considered a recommended choice due to scalability. Public clouds are also favored in the organization domain, because a public cloud infrastructure will minimize the number of partners required to deliver the service.

In the financial domain three main issues can be seen. First, a monthly fee based on API hits per month is the main revenue model for free APIs, which form the majority of APIs used on the Internet. Commissions on the fee based web services cannot be the only revenue model as it is only a niche market. Second, low integration costs of API providers can be more important than the operational costs of the system. Third, by applying a public cloud infrastructure, the investment risks can be reduced and operational costs can be predicted.

The results from the API management case study can be directly applied to the GSMA OneAPI system. In the early phases API service offerings will be limited, but future extensions will give developers longer term trust in the system. Existing solutions should be reused, to avoid high investment costs. Cloud technologies offer an interesting alternative to lower both the capex and opex costs. API technologies must support both individual and enterprise developers.

The broker model is not the only option for a cross network service provider offering. Operators may also provide APIs directly to the developers. This model can be called a virtual broker. The model is depicted in figure 7. As an advantage, the system does not require a new actor or point of failure into the value network. As a drawback, usage of APIs becomes more complicated, because developers must identify the correct operator of the user. Due to number portability feature, the number prefix does not anymore reveal the correct operator, but user’s operator must be verified using a number translation service.
A virtual broker enables an innovation space described in figure 3 [14]. It supports two-sided business models [16, 17], as well as a single point of contract between operators and users, developers, and service providers. Moreover, a virtual broker provides a simultaneously cooperative and competitive business environment where cooperation maximizes network externalities and competition assures effectiveness of the environment. At later phases, when markets are becoming mature, a virtual broker can be replaced with one or several physical brokers.

Financially a virtual broker can support both pay-per-use and subscription models. Transparent pricing enables true competition between the operators. Standardized interfaces lower the integration costs, and avoidance of the broker actor reduces the players in revenue sharing. Both private and public cloud infrastructures can be used to implement the operator service delivery platforms.

6. Conclusions

The paper introduced a case study on business models of internet API management providers and the GSMA Canadian OneAPI pilot. An analysis was reviewed based on step 1, the Quick Scan, of the STOF (service, technology, organization and finance) method. The STOF framework balances various requirements in a coherent way. The literature review supports the trend in increasing the number of open APIs. Moreover, industry trends prove that openness is becoming a de facto in the telecommunications sector, too. The internet API management providers bring useful insights to the Open Telco initiative. The Open Telco concept is suitable for developed markets known for advanced mobile infrastructures and saturated voice and text messaging services. As a conclusion the paper proposes a virtual broker model that fulfils the presented requirements in an optimal way. Future research should apply the latter steps of the STOF method to a live GSMA OneAPI pilot, implemented in private or public cloud infrastructures.

7. Acknowledgements

We thank the anonymous reviewers for their valuable comments. The work is supported by Tekes (the Finnish Funding Agency for Technology and Innovation, [www.tekes.fi]) as a part of the Cloud Software Program ([www.cloudsoftwareprogram.org]) of Tivit (Strategic Centre for Science, Technology and Innovation in the Field of ICT, [www.tivit.fi]).

8. References


